Amendments to the Drawings:

Please replace Figure 4, which includes an arrow to the right of the "DECODER PROGRAM" block, with the attached Replacement Sheet of Figure 4, which includes a "START" oval, an "END" oval, a "PERFORM OPTIMIZATION?" decision block 415 between blocks 414 and 420, and an "OPTIMIZATION COMPLETE?" decision block 426 between block 424 and the "END" oval. Also, the arrow to the right of the "DECODER PROGRAM" block has been deleted in the attached Replacement Sheet of Figure 4.

Attachment:	Replacement Sheet
	Annotated Sheet showing changes

REMARKS/ARGUMENTS

Amendments are made to the specification. Support for the amendments to the specification can be found in Applicants' patent application on page 5, line 3 – page 6, line 2 and page 11, lines 6-12.

Claims 1-20 are pending in the present application. Claims 17-20 are cancelled. Claims 1, 10, and 11 are amended. Claims 21-24 are added. Support for the new claims and claim amendments can be found in the claims as originally filed and in the Applicants' patent application on page 9, lines 10-18 and page 11, lines 1-5. Reconsideration of the claims is respectfully requested.

I. Interview Summary

Applicants thank the Examiner for the interview held on May 2, 2007 between the Applicants' representatives and the Examiner. The rejection of claim 1 under 35 U.S.C. § 102, and the rejection of claim 11 under 35 U.S.C. § 101 was discussed.

Regarding the rejection of claim 1, an agreement was reached that the amendment to claim 1 would overcome the 35 U.S.C. § 102 rejection. Regarding the rejection of claim 11, an agreement was reached that the amendment to claims 11 would overcome the 35 U.S.C. § 101 rejection. An agreement was also reached that no 35 U.S.C. § 101 rejection could be made against new claim 21.

II. 35 U.S.C. § 101

The Examiner rejected claims 11-20 as directed towards non-statutory subject matter. Regarding claims 17-20, the rejection is now moot as claims 17-20 have been cancelled. Regarding claims 11-16, Applicants have amended claims 11-16 accordingly, thereby overcoming the rejection.

III. Objections to the Claims

The Examiner objected to claims 17 and 20 as containing informalities. Claims 17 and 20 are cancelled. Therefore, the objections are now moot.

IV. Objection to the Drawings

The Examiner objected to the drawings. In response, Applicants have submitted a Replacement Drawing with this Response which includes a "START" oval, an "END" oval, a "PERFORM OPTIMIZATION?" decision block 415 between blocks 414 and 420, and an "OPTIMIZATION COMPLETE?" decision block 426 between block 424 and the "END" oval. Applicants have also submitted an annotated sheet of original Figure 4 illustrating the changes to the drawing. Therefore, the objection should be overcome. No new matter has been added. Support for these changes can be found on page 5, line 3 – page 6, line 2 and page 11, lines 6-12.

V. <u>35 U.S.C. § 102, Anticipation; Claim 1-4, 6, and 11-13</u>

The Examiner rejected claims 1-4, 6, and 11-13 as anticipated by *Arnold et al.*, <u>Method and Apparatus for Providing a Dynamic Speech-Driven Control and Remote Service Access System</u>, U.S. Patent Application Publication 2003/0125955, December 28, 2001 (hereinafter "*Arnold*"). The rejection is respectfully traversed. With regard to claim 1, the Examiner states:

> *Arnold* discloses a method for generating task-specific code for pattern recognition ([0008]), the method comprising:
> receiving task-specific input system data of a pattern recognition system and generating task-specific code for the pattern recognition. system based on the task-specific input system data ("the distributed speech recognition system allows automatic "speaker adaptation" to be performed locally by the client device . . . local parameters . . . are adapted locally by the client device in performing its speech recognition function", [0008], see also [0009] and "the client device is adapting its models in response to the speaker", [0020])

Office Action dated February 13, 2007, p. 5.

A prior art reference anticipates the claimed invention under 35 U.S.C. § 102 only if every element of a claimed invention is identically shown in that single reference, arranged as they are in the claims. *In re Bond,* 910 F.2d 831, 832, 15 U.S.P.Q.2d 1566, 1567 (Fed. Cir. 1990). All limitations of the claimed invention must be considered when determining patentability. *In re Lowry*, 32 F.3d 1579, 1582, 32 U.S.P.Q.2d 1031, 1034 (Fed. Cir. 1994). Anticipation focuses on whether a claim reads on the product or process a prior art reference discloses, not on what the reference broadly teaches. *Kalman v. Kimberly-Clark Corp.*, 713 F.2d 760, 218 U.S.P.Q. 781 (Fed. Cir. 1983). In this case, every feature of the presently claimed invention is not identically shown in the cited reference, arranged as they are in the claims.

Amended claim 1 is representative of claim 11. Amended claim 1 is as follows:

> 1. A method, implemented in a data processing system, for generating task-specific code for pattern recognition, the method comprising:
> receiving task-specific input system data of a pattern recognition system; and
> generating task-specific code for the pattern recognition system based on the task-specific input system data, wherein the task-specific code includes computer language suitable for compilation.

Arnold does not anticipate claim 1 because *Arnold* does not disclose each and every feature of claim 1 as amended. Specifically, *Arnold* fails to disclose the feature of generating task-specific code for the pattern recognition system based on the task-specific input system data, wherein the task-specific code includes computer language suitable for compilation.

The Examiner cites various portions of *Arnold*. Each of these portions will be addressed in turn to show that *Arnold* does not anticipate claim 1 as amended. Applicants first address the following portion of *Arnold*:

> [0008] Specifically, the distributed speech recognition system comprises at least one client device and a central server. The client device is a remote device that is in communication with the central server, but it is physically deployed apart from the central server. In operation, the client device is equipped with a speech recognition module having an initial language model. In one embodiment, the distributed speech recognition system allows automatic "speaker adaptation" to be performed locally by the client device. Namely, local parameters such as environmental noise around the speaker, pronunciation (e.g., accents or dialect) of the speaker and/or the acoustic environment (e.g., within a tunnel or a carpeted room) around the speaker are adapted locally by the client device in performing its speech recognition functions. Speaker adaptation is particularly appropriate within the present architecture in that it is carried out in a client device largely dedicated to a particular user. Although such adaptations are performed locally, the central server may also assist the client device as necessary, e.g., forwarding a different acoustic model to the client device from the central server.

Arnold, paragraph 8.

Neither the cited portion nor any other portion of *Arnold* discloses the feature of generating task-specific code for the pattern recognition system based on the task-specific input system data, wherein the task-specific code includes computer language suitable for compilation. *Arnold* discloses a speech recognition and processing system for a client device in a client/server architecture that can be dynamically and remotely altered depending on the user. A language model in *Arnold's* client device may be updated dynamically by migrating language models from the server device, or by updating local parameters. However, *Arnold* nowhere discloses generating task-specific code that includes computer language suitable for compilation.

On the other hand, claim 1 recites the feature of generating task-specific code for the pattern recognition system based on the task-specific input system data, wherein the task-specific code includes computer language suitable for compilation. The cited portion of *Arnold* differs from the claimed feature because the cited portion nowhere discloses task-specific code that includes computer language suitable for compilation, let alone generating task-specific code that includes computer language suitable for compilation. For example, the cited portion states that "local parameters such as environmental noise around the speaker, pronunciation (e.g., accents or dialect) of the speaker and/or the acoustic environment (e.g., within a tunnel or a carpeted room) around the speaker are adapted locally by the client device in performing its speech recognition functions." As a first matter, the cited statement discloses only "adapting" local parameters, but nowhere discloses that local parameters are generated, as claimed. As a second matter, even assuming, *arguendo*, that local parameters are generated, the cited statement still discloses nothing related to whether the local parameters are suitable for compilation because the cited statement does not relate to compilation or suitability for compilation at all. In fact, *Arnold* nowhere discloses compiling anything, and the term "compile" or any equivalent word fails to appears anywhere in *Arnold*. Therefore, *Arnold* fails to disclose the feature of generating task-specific code for the pattern recognition system based on the task-specific input system data, wherein the task-specific code includes computer language suitable for compilation.

Next, the Examiner cites the following portion of *Arnold*:

> [0009] For example, in another embodiment, the distributed speech recognition system provides the ability to implement "dynamic grammars". Specifically, the client device is initially equipped with an initial language model. As the user interacts with the client device, the language model is updated by the central server as the interactions between the user and client device indicate that an updated language model is required to carry out the user's request. This distributed approach maximizes the processing power of the client device without overburdening the client device unnecessarily with a complex language model.

Arnold, paragraph 9.

Neither the cited portion nor any other portion of *Arnold* discloses the feature of generating task-specific code for the pattern recognition system based on the task-specific input system data, wherein the task-specific code includes computer language suitable for compilation. The cited portion describes how *Arnold's* system updates an initial language model in a client device to reduce the burden on a client device. However, the cited portion does not disclose that anything in the cited portion is suitable for compilation, as claimed, let alone task-specific code that includes computer language suitable for compilation. For example, the cited portion states that "[a]s the user interacts with the client device, the language model is updated by the central server as the interactions between the user and client device indicate that an updated language model is required to carry out the user's request." However, the cited statement discloses only that a language model is updated by a central server, but nowhere discloses whether the language model is suitable for compilation. In fact, the cited statement does not relate to compilation or suitability for compilation at all. Because the cited statement does not relate to whether the language model is suitable for compilation or compilation in general, the cited statement fails to disclose the feature of generating task-specific code for the pattern recognition system based on the task-specific input system data, wherein the task-specific code includes computer language suitable for compilation.

The cited portion also shows *Arnold's* lack of any reason to disclose generating task-specific code that includes computer language suitable for compilation because *Arnold* has no reason to compile any code. *Arnold's* system does not compile any new language models or programs from a source code level, and instead updates and adapts language models that already exist on a client or server. Because *Arnold* does not compile any new language models or programs from a source code level, *Arnold* has no reason to disclose task-specific code that includes computer language suitable for compilation. Indeed, as mentioned above, *Arnold* nowhere mentions the term "compile" or any equivalent term. Because *Arnold* has no reason to disclose task-specific code that includes computer language suitable for compilation, *Arnold* has no reason to disclose generating task-specific code that includes computer language suitable for compilation. Therefore, *Arnold* fails to disclose, and has no reason to disclose, the feature of generating task-specific code for the pattern recognition system based on the task-specific input system data, wherein the task-specific code includes computer language suitable for compilation.

Lastly, the Examiner cites the following portion of *Arnold*:

> [0020] In turn, the speech recognizer 120 receives the speech features and is able to decode the "recognized text" from the speech features using various models as discussed below. An important aspect of the present invention pertains to the "dynamic" models that are employed by the speech recognizer 120. Specifically, due to the anticipated small footprint of the client device 110, the present invention employs "dynamic grammars" as a driving mechanism in providing the necessary models or portions or updates of a model to the client device. Since the processing capability and storage capability of the client device 110 are anticipated to be limited, the present invention is designed to provide the client device 110 with just enough data and information to perform the tasks as required by a current speaker. Thus, the client device is adapting its models in response to the speaker, hence the term "dynamic grammars". The speaker adaptation functions are executed in cooperation with the central server 130.

Arnold, paragraph 20.

Neither the cited portion nor any other portion of *Arnold* discloses the feature of generating task-specific code for the pattern recognition system based on the task-specific input system data, wherein the task-specific code includes computer language suitable for compilation. The cited portion discloses decoding speech using models, as well as adapting and updated language models on a client device. However, the cited portion differs from the claimed feature because the cited portion discloses nothing related to the compilation of language models or decoded speech. For example, the cited portion states that "the speech recognizer 120 receives the speech features and is able to decode the 'recognized text' from the speech features using various models as discussed below." However, the cited statement discloses only decoding recognized text from speech using various models, but nowhere discloses whether the decoded speech or various models are suitable for compilation. In fact, the cited statement does not relate to compilation or suitability for compilation at all.

As another example, the cited portion states that "due to the anticipated small footprint of the client device 110, the present invention employs "dynamic grammars" as a driving mechanism in providing the necessary models or portions or updates of a model to the client device." However, the cited statement discloses only updating and providing models to a client device. As in the previously cited statement, the cited statement at issue does not relate to compilation or suitability for compilation at all, and does not address whether dynamic grammars or models are suitable for compilation. Because neither the cited portion, nor any other portion, of *Arnold* addresses compilation or suitability for compilation, and because *Arnold* has no need to address compilation, *Arnold* fails to disclose, and has no reason to disclose, the feature of generating task-specific code for the pattern recognition system based on the task-specific input system data, wherein the task-specific code includes computer language suitable for compilation. Accordingly, *Arnold* fails to disclose all of the features of claim 1.

Because claim 1 is representative of claim 11, the same distinctions between claim 1 and the reference also applies to claim 11. Additionally, because claims 2-4, 6, 12, and 13 depend from claims 1 and 11, at least the same distinctions between *Arnold* and claims 1 and 11 apply for these claims as well. Additionally, claims 2-4, 6, 12, and 13 claim other additional combinations of features not disclosed by the reference. Therefore, the rejection of claims 1-4, 6, and 11-13 under 35 U.S.C. § 102 has been overcome.

VI. <u>35 U.S.C. § 103, Obviousness; Claim 5</u>

The Examiner rejected claim 5 under 35 U.S.C. § 103 as obvious over *Arnold* in view of *IBM Technical Disclosure Bulletin*, <u>Determining the Probability of Words in a String with a Word-Skipping Model</u>, November 1, 1985 (hereinafter "*IBM Bulletin*"). This rejection is respectfully traversed.

The Examiner states:

> *Arnold* discloses the method of claim 3, but does not disclose wherein the language model is represented as a Hidden Markov Model ([0023]).
>
> The IBM Technical Disclosure Bulletin discloses a speech recognition method where the language model is "defined as a Markov source (a hidden Markov chain)" (page 2, lines 1 1-1 3).
>
> Therefore it would have been obvious to one with ordinary skill in the art at the time of the invention to represent the language model as a Hidden Markov Model in *Arnold's* method because off the shelf Hidden Markov Model software was available therefore freeing *Arnold* from programming another model.

Office Action dated February 13, 2007, p. 5.

No *prima facie* obviousness rejection can be made against claim 5 because neither *Arnold* nor *IBM Bulletin* teaches or suggests all of the features of claim 5. Further, *Arnold* and *IBM Bulletin* may not be properly combined to achieve the invention of claim 5.

VI.A. Neither *Arnold* nor *IBM Bulletin* Teaches or Suggests All of the Features of Claim 5

No *prima facie* obviousness rejection can be stated because neither reference, alone or in combination, teaches all of the features of claim 5. The Examiner bears the burden of establishing a *prima facie* case of obviousness based on the prior art when rejecting claims under 35 U.S.C. § 103. *In re*

Fritch, 972 F.2d 1260, 23 U.S.P.Q.2d 1780 (Fed. Cir. 1992). Additionally, all limitations of the claimed invention must be considered when determining patentability. *In re Lowry*, 32 F.3d 1579, 1582, 32 U.S.P.Q.2d 1031, 1034 (Fed. Cir. 1994). Therefore, the Examiner fails to state a *prima facie* obviousness rejection if the proposed combination does not teach all of the features of the claimed invention.

No *prima facie* obviousness rejection can be made against claim 5 because neither *Arnold* nor *IBM Bulletin* teaches or suggests all of the features of claim 5. Claim 5 is as follows:

> 5. The method of claim 3, wherein the language model is represented as a Hidden Markov Model.

A *prima facie* obviousness rejection cannot be made because the proposed combination of the references does not teach or suggest all of the features of claim 5. As shown above, *Arnold* does not teach or suggest all of the features of claim 1. Therefore, *Arnold* does not teach or suggest all of the features of claim 5, which depends from claim 1.

IBM Bulletin does not cure *Arnold's* lack of disclosure. *IBM Bulletin* discloses a language model that assigns probabilities to strings of words by skipping words. However, *IBM Bulletin* fails to teach or suggest any of the features of claim 1, and the Examiner does not assert otherwise. Therefore, the proposed combination of *Arnold* and *IBM Bulletin*, when considered as a whole, does not teach or suggest all of the features of claim 5, which depends from claim 1. Accordingly, no *prima facie* obviousness rejection can be made against claim 5 and the rejection of claim 5 under 35 U.S.C. § 103 has been overcome.

VI.B. The Examiner Failed to State a Proper Teaching, Motivation, or Suggestion to Combine the References

In addition, no *prima facie* obviousness rejection can be made against claim 5 because no proper teaching or suggestion to combine the references has been stated. A *prima facie* case of obviousness is established when the teachings of the prior art itself suggest the claimed subject matter to a person of ordinary skill in the art. *In re Bell*, 991 F.2d 781, 783, 26 U.S.P.Q.2d 1529, 1531 (Fed. Cir. 1993). A proper *prima facie* case of obviousness cannot be established by combining the teachings of the prior art absent some teaching, incentive, or suggestion supporting the combination. *In re Napier*, 55 F.3d 610, 613, 34 U.S.P.Q.2d 1782, 1784 (Fed. Cir. 1995); *In re Bond*, 910 F.2d 831, 834, 15 U.S.P.Q.2d 1566, 1568 (Fed. Cir. 1990). No such teaching or suggestion is present in the cited references and the Examiner has not pointed out any proper teaching or suggestion that is based on the prior art.

In the case at hand, the Examiner has stated only a proposed advantage to combining the references. Specifically, the Examiner states that:

> Therefore it would have been obvious to one with ordinary skill in the art at the time of the invention to represent the language model as a Hidden Markov Model

> in *Arnold's* method because off the shelf Hidden Markov Model software was available therefore freeing *Arnold* from programming another model.

Office Action dated February 13, 2007, p. 5.

However, an advantage is not necessarily a teaching, suggestion, or motivation. Further, the references themselves do not suggest the proposed advantage. In the present case, neither *Arnold* nor *IBM Bulletin* teach using *IBM Bulletin*'s hidden Markov chain as a language model in *Arnold's* remotely adaptable speech recognition and processing system. Accordingly, the Examiner has not actually stated a teaching, motivation, or suggestion based on the references to combine the references.

In addition, the proposed advantage of "freeing *Arnold* from programming another model" does not actually exist because the proposed advantage does not benefit *Arnold's* remotely adaptable speech recognition and processing system. *Arnold's* system migrates existing language models from a server to a client device and updates the language models on the client device as needed. *Arnold* nowhere discloses programming entirely new language models, and instead focuses on alleviating the burden on a client device by migrating and updating existing models.

For example, *Arnold* discloses that "the client device is initially equipped with an initial language model. As the user interacts with the client device, the language model is updated by the central server as the interactions between the user and client device indicate that an updated language model is required to carry out the user's request." *Arnold*, paragraph 9. Neither the cited statement nor any other portion of *Arnold* pertains to programming entirely new language models. Because *Arnold's* system does not pertain to programming new language models, and instead pertains to migrating and updating existing language models on servers and client, the Examiner's proposed advantage of "freeing *Arnold* from programming another model" would not actually benefit *Arnold's* system. Because the Examiner's proposed advantage of "freeing *Arnold* from programming another model" would not benefit *Arnold's* system, the Examiner's proposed advantage does not actually exist. In the absence of any other pre-existing teaching, suggestion, or motivation to combine the references, which the Examiner has not presented, no pre-existing teaching, suggestion, or motivation to combine the references exists. Accordingly, no *prima facie* obviousness rejection can be stated against claim 5.

VII. 35 U.S.C. § 103, Obviousness; Claims 7, 14, 17-19, and New Claims 21, 22, and 23

The Examiner rejected claims 7, 14, and 17-19 under 35 U.S.C. § 103 as obvious over *Arnold* in view of *Poirier et al.*, Executable for Requesting a Linguistic Service, U.S. Patent 6,321,372, December 23, 1998 (hereinafter "*Poirier*"). Claims 17-19 are cancelled. Therefore, the rejection of claims 17-19 is now moot. New claims 21, 22, and 23 have been added. New claims 21, 22, and 23 contain features similar to claims 17 and 18, respectively. This rejection is respectfully traversed with respect to these new claims. The Examiner states that:

Arnold discloses the method of claim 1, but does not explicitly disclose compiling the task-specific code to form a decoder program.

Poirier discloses a similar method where a source code is modified, such as by further specifying it, in a linguistic service system. *Poirier* also discloses the code being compiled ("compile . . . modified source code", col. 10, lines 52-56).

It would have been obvious to one with ordinary skill in the art at the time the invention was made to compile a source code in order to obtain an executable file which is able to run on a computer (*Poirier*, "compile ... modified source code to obtain service executable", col. 10, lines 52-56).

Claim 14 is similar in scope and content to claim 7 and is rejected with the same rationale.

Claim 17 is similar in scope and content to claim 7 and is rejected with the same rationale.

Office Action dated February 13, 2007, pp. 7-8.

No *prima facie* obviousness rejection can be made against claims 7, 14, 21, 22, and 23 because neither *Arnold* nor *Poirier* teaches or suggests all of the features of claims 7, 14, 21, 22, and 23. Further, *Arnold* and *Poirier* may not be properly combined to achieve the invention of claims 7, 14, 21, 22, and 23.

VII.A. Neither *Arnold* nor *Poirier* Teaches or Suggests All of the Features of Claims 7, 14, 21, 22, and 23

A *prima facie* obviousness rejection cannot be made because the proposed combination of the references does not teach or suggest all of the features of claims 7, 14, 21, 22, and 23. As shown above, *Arnold* does not teach or suggest all of the features of claim 1. Because claim 21 contains similar features as claim 1, *Arnold* does not teach or suggest all of the features of claim 21. Specifically, *Arnold* fails to teach or suggest the feature of instructions to generate task-specific code for the pattern recognition system based on the task-specific input system data, wherein the task-specific code includes computer language suitable for compilation, which is recited in claim 21.

Poirier does not cure *Arnold's* lack of disclosure. *Poirier* discloses hierarchal linguistic services in which descendent linguistic services may be programmed based on an ancestral linguistic service. However, *Poirier* fails to teach or suggest any of the features of claim 1, and the Examiner does not assert otherwise. Therefore, the proposed combination of *Arnold* and *Poirier*, when considered as a whole, does not teach or suggest all of the features of claim 21, which contains similar features as claim 1. Accordingly, no *prima facie* obviousness can be made against claim 21 and the rejection of claim 21 under 35 U.S.C. § 103 has been overcome.

Claim 21 also contains additional features not taught or suggested by *Arnold* or *Poirier*. Claim 21, which is representative of claims 7 and 14, is as follows:

> 21. An apparatus for generating task-specific code for pattern recognition, the apparatus comprising:
>
> a bus;
>
> a memory connected to the bus, wherein the memory contains computer readable instructions; and
>
> a processor connected to the bus, wherein the processor executes the computer readable instructions to:
>
> receive task-specific input system data of a pattern recognition system;
>
> generate task-specific code for the pattern recognition system based on the task-specific input system data, wherein the task-specific code includes computer language suitable for compilation; and
>
> compile the task-specific code to form a decoder program for the pattern recognition system.

Specifically, *Arnold* and *Poirier*, considered together as a whole, fail to teach or suggest the feature of instructions to compile the task-specific code to form a decoder program for the pattern recognition system. Nonetheless, the Examiner cites the following portion of *Poirier*:

> The operations requested by user signals can include editing operations that modify preexisting source code 100 or intermediate versions to obtain modified source code 124 and also compile or interpret operations that use modified source code 124 to obtain service executable 126.

Poirier, col. 10, ll. 52-56.

Neither the cited portion nor any other portion of *Poirier* teaches or suggests the feature of instructions to compile the task-specific code to form a decoder program for the pattern recognition system. *Poirier* discloses hierarchal linguistic services in which descendent linguistic services may be programmed based on an ancestral linguistic service. The cited portion discloses user requests to compile operations that use modified source code. However, the compiling operations that use modified source code is not the same as instructions to compile the task-specific code to form a decoder program because *Poirier* fails to teach or suggest task-specific code.

On the other hand, claim 21 recites the feature of instructions to compile the task-specific code to form a decoder program for the pattern recognition system. Claims 21 further defines the feature of task-specific code as follows: generate task-specific code for the pattern recognition system based on the task-specific input system data, wherein the task-specific code includes computer language suitable for compilation. However, as shown above, neither *Arnold* nor *Poirier* teach or suggest the feature of instructions to generate task-specific code for the pattern recognition system based on the task-specific input system data, wherein the task-specific code includes computer language suitable for compilation. Therefore, neither *Arnold* nor *Poirier* teach or suggest "task-specific code" as defined by claim 21. Because neither *Arnold* nor *Poirier* teach or suggest "task-specific code," neither reference can teach or suggest instructions to compile the task-specific code to form a decoder program for the pattern recognition system, as claimed in claim 21. Therefore, the proposed combination of *Arnold* and *Poirier*, when considered as a whole, does not teach or suggest all of the features of claim 21. Accordingly, no *prima facie* obviousness rejection can be made against claim 21.

Because claim 21 is representative of claims 7 and 14, the same distinctions between the references and the claimed invention in claim 21 can also be made for claims 7 and 14. Also, because claims 22 and 23 depend from claim 21, at least the same distinctions between the references and the claimed invention of claim 21 can also be made for claims 22 and 23. Accordingly, no *prima facie* obviousness rejections can be made against these claims for the reasons presented above.

VII.B. The Examiner Failed to State a Proper Teaching, Motivation, or Suggestion to Combine the References

No *prima facie* obviousness rejection can be made against claims 7, 14, 21, 22, and 23 because no proper teaching or suggestion to combine the references has been stated. The Examiner has stated only a proposed advantage to combining the references. Specifically, the Examiner states that:

> It would have been obvious to one with ordinary skill in the art at the time the invention was made <u>to compile a source code in order to obtain an executable file which is able to run on a computer</u> (*Poirier*, "compile ... modified source code to obtain service executable", col. 10, lines 52-56).

Office Action dated February 13, 2007, p. 7 (emphasis added).

However, an advantage is not necessarily a teaching, suggestion, or motivation. Further, the references themselves do not suggest the proposed advantage. In the present case, neither *Arnold* nor *Poirier* teach using *Poirier's* hierarchal language services in *Arnold's* remotely adaptable speech recognition and processing system. Accordingly, the Examiner has not actually stated a teaching, motivation, or suggestion based on the references to combine the references.

In addition, the proposed advantage of being able to "compile a source code in order to obtain an executable file which is able to run on a computer" does not actually exist because the proposed advantage does not benefit *Arnold's* remotely adaptable speech recognition and processing system. *Arnold* nowhere addresses the source code level of the system disclosed therein, and nowhere discloses any source code, let alone compiling source code. Also, as shown in Section V, *Arnold* fails to disclose task-specific code that includes computer language suitable for compilation. Because *Arnold* discloses no source code suitable for compilation, the ability to "compile a source code in order to obtain an executable file which is able to run on a computer" does not benefit *Arnold.* In other words, *Arnold* cannot benefit from compiling source code because *Arnold* discloses no source code suitable for compilation. Because the Examiner's proposed advantage of being able to "compile a source code in order to obtain an executable file which is able to run on a computer" would not benefit *Arnold's* system, the Examiner's proposed advantage does not actually exist. In the absence of any other pre-existing teaching, suggestion, or motivation to combine the references, which the Examiner has not presented, no pre-existing teaching, suggestion, or motivation to combine the references exists. Accordingly, the Examiner no *prima facie* obviousness rejection can be made against these claims.

VIII. 35 U.S.C. § 103, Obviousness; Claims 8-10, 15, 16, and 20

The Examiner rejected claims 8-10, 15, 16, and 20 under 35 U.S.C. § 103 as obvious over *Arnold* in view of *Poirier* and in further view of *Lanning*, Apparatus and Method for Optimizing Applications for Multiple Operational Environments or Modes, U.S. Patent 5,787,285, August 15, 1995 (hereinafter "*Lanning*"). Claim 20 is cancelled. Therefore, the rejection of claim 20 is now moot. New claim 24 has been added. New claim 24 contains features similar to cancelled claim 20. This rejection is respectfully traversed. The Examiner states:

> *Arnold* and *Poirier* disclose the method of claim 7, but they do not explicitly disclose profiling the decoder program to form a profile and determining whether the decoder program is optimized.
>
> *Lanning* discloses a method of optimizing executable software where the code is profiled and optimized ("automated "profilers" to provide data to these optimizing compilers", col. 1, lines 42-52).
>
> It would have been obvious to one with ordinary skill in the art at the time the invention was made to profile and optimize the code in *Arnold* and *Poirier's* method in order to "enhance the run-time performance of a piece of software (*Lanning*, col. 1, lines 42-52).

Office Action dated February 13, 2007, pp. 8-9.

No *prima facie* obviousness rejection can be made against claims 8-10, 15, 16, and 24 because *Arnold*, *Poirier*, and *Lanning* fail to teach or suggest all of the features of claims 8-10, 15, 16, and 24. Further, *Arnold*, *Poirier*, and *Lanning* may not be properly combined to achieve the invention of claims 8-10, 15, 16, and 24.

VIII.A. *Arnold*, *Poirier*, and *Lanning* Fail to Teach or Suggest All of the Features of Claims 8-10, 15, 16, and 24

A *prima facie* obviousness rejection cannot be made because the proposed combination of the references does not teach or suggest all of the features of claims 8-10, 15, 16, and 24. As shown in Sections VII.A., neither *Arnold* nor *Poirier* teach or suggest all of the features of claims 1 and 21. Because claim 1 is representative of claim 11, the same distinctions between the references and the claimed invention in claim 1 can also be made for claim 11. Therefore, neither *Arnold* nor *Poirier*, alone or in combination, teach or suggest all of the features of claims 1, 11, and 21.

Lanning does not cure *Arnold* and *Poirier's* lack of disclosure. *Lanning* discloses a method for optimizing applications for multiple operational environments or modes. However, *Lanning* fails to teach or suggest any of the features of claims 1, 11, and 21, and the Examiner does not assert otherwise. Therefore, the proposed combination of *Arnold, Poirier*, and *Lanning*, when considered as a whole, does not teach or suggest all of the features of claims 1, 11, and 21. Because *Arnold, Poirier*, and *Lanning*, alone or in combination, fail to teach or suggest all of the features of claims 1, 11, and 21, no *prima facie* obviousness rejection can be made against claims 8-10, 15, 16, and 24, at least by virtue of their dependency on claims 1, 11, and 21.

Claim 8 also contains additional features that are not taught by *Arnold, Poirier*, and *Lanning*. Claim 8, representative of claims 15 and 24, is as follows:

> 8. The method of claim 7, further comprising:
> profiling the decoder program to form a profile; and
> determining whether the decoder program is optimized.

Specifically, *Arnold, Poirier*, and *Lanning*, considered as a whole, fail to teach or suggest the feature of determining whether the decoder program is optimized. Nonetheless, the Examiner asserts the following portion of *Lanning*:

> Automated tools such as optimizing compilers are now commercially available to enhance the run-time performance of a piece of software. Recent technical papers have also disclosed the use of automated "profilers" to provide data to these optimizing compilers. In general, these profilers monitor the execution of a particular piece of software and gather information on the most likely paths of execution. The optimizing compiler then uses the information gathered by the profiler to recompile the source code in a more efficient manner.

Lanning, col. 1, ll. 42-52.

Neither the cited portion nor any other portion of *Lanning* teaches or suggests the feature of determining whether the decoder program is optimized. *Lanning* discloses a method for optimizing applications for multiple operational environments or modes. The cited portion describes the function of optimizing compilers and profilers. For example, the profilers disclosed in the cited portion monitor execution and gather execution path information. However, the cited portion nowhere discloses that the optimizing compilers or the profilers make a determination as to whether a decoder program is optimized.

On the other hand, claim 8 recites the feature of determining whether the decoder program is optimized. The cited portion differs from the claimed feature because the cited portion does not disclose making any determination, let alone determining whether a program is optimized. For example, the cited portion states that "profilers monitor the execution of a particular piece of software and gather information on the most likely paths of execution." However, the cited statement discloses only that profilers monitor execution and gather execution path information, and does not disclose any determination whatsoever, let alone determining whether a program is optimized.

The cited portion also states that "[a]utomated tools such as optimizing compilers are now commercially available to enhance the run-time performance of a piece of software." However, enhancing the run-time performance of a piece of software, as disclosed in the cited statement, is not the same as determining whether a program is optimized because the cited statement nowhere teaches or suggests that enhancing performance involves determining whether a program is optimized.

The failure of *Lanning* to teach or suggest the feature of determining whether the decoder program is optimized is expected because *Lanning* does not teach or suggest discriminating between programs based on whether the program is optimized before implementing the method disclosed therein. Instead, *Lanning* discloses a method that is used on any program regardless of whether the program is optimized. For example, *Lanning* provides:

> Briefly, the present invention optimizes an executable software program containing a plurality of basic blocks for several different operational environments or operational modes by identifying the basic blocks which execute for each particular operational environment or operational mode, and the frequency of each blocks' execution in the environment or mode. For each environment or mode, the frequency of execution for the block in that environment or mode is compared against a predetermined threshold value. Each basic block whose frequency of execution exceeds the predetermined threshold value is copied into a program segment for that environment or mode. Basic blocks whose frequency of execution does not exceed the predetermined threshold value are copied into a common code segment accessible from each of the program segments. The code in each program segment is then optimized.

Lanning, col. 2, ll. 45-60.

The quoted portion discloses that "the present invention optimizes an executable software program," but does not address whether the executable software program is optimized in the first place. Because *Lanning's* method is used on any program regardless of whether the program is optimized, *Lanning* has no need to determining whether a program is optimized. Therefore, *Lanning* fails to disclose, and has no reason to disclose, the feature of determining whether the decoder program is optimized.

Neither *Arnold* nor *Poirier* cure *Lanning's* lack of disclosure. Indeed, the Examiner agrees that "*Arnold* and *Poirier* ... do not explicitly disclose profiling the decoder program to form a profile and determining whether the decoder program is optimized." Also, *Arnold* and *Poirier* fail to suggest the claimed feature. Therefore, the proposed combination of *Arnold*, *Poirier*, and *Lanning*, when considered as a whole, fails to teach or suggest all of the features of claim 8.

Because claim 8 is representative of claims 15 and 24, the same distinctions between claim 8 and the references also apply to claims 15 and 24. Because of *Arnold, Poirier*, and *Lanning*, alone or in combination, fail to teach or suggest all of the features of claims 8 and 15, no *prima facie* obviousness rejection can be made against claims 9, 10, and 16, at least by virtue of their dependency on claims 8 and 15. Therefore, no *prima facie* obviousness rejection can be stated against claims 8-10, 15, 16, and 24.

VIII.B. No Proper Teaching, Motivation, or Suggestion Exists to Combine the References

The rejection of claims 8-10, 15, 16, and 24 relies on the incorrect premise that *Arnold* and *Poirier* can be combined in the manner the Examiner proposed vis-à-vis claims 7, 14, 21, 22, and 23. As shown in Section VII.B., the Examiner failed to state a proper teaching, motivation, or suggestion to combine *Arnold* and *Poirier* to achieve the inventions of claims 7, 14, 21, 22, and 23. The rejection of claims 8-10, 15, 16, and 24 does not address the deficiencies in the rejection of claims 7, 14, 21, 22, and 23. Therefore, no *prima facie* obviousness rejection can be stated against claims 8-10, 15, 16, and 24.

IX. Conclusion

The subject application is patentable over the cited references and should now be in condition for allowance. The Examiner is invited to call the undersigned at the below-listed telephone number if in the opinion of the Examiner such a telephone conference would expedite or aid the prosecution and examination of this application.

DATE: 05/14/07

Respectfully submitted,

/Theodore D. Fay III/

Theodore D. Fay III
Reg. No. 48,504
Yee & Associates, P.C.
P.O. Box 802333
Dallas, TX 75380
(972) 385-8777
Attorney for Applicants

TF/ka